Titanic Predictions

John Vithoulkas

The following project was inspired by Kaggle's Machine Learning competition. The competition can be viewed here.

I certify on my honor that this is my own, original work.

Topics Covered:

- Exploratory Data Analysis
- Visualizations using GGplot
- Logistic Regression
- Random Forests

Exploratory

Let's take a look into the data.

```
str(train)
```

```
## 'data.frame':    891 obs. of  12 variables:
##  $ PassengerId: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Survived   : int  0 1 1 1 0 0 0 0 1 1 ...
##  $ Pclass     : int  3 1 3 1 3 3 1 3 3 2 ...
##  $ Name       : chr  "Braund, Mr. Owen Harris" "Cumings, Mrs. John Bradley (Florence Briggs Thayer)"
##  $ Sex        : chr  "male" "female" "female" "female" ...
##  $ Age        : num  22 38 26 35 35 NA 54 2 27 14 ...
##  $ SibSp      : int  1 1 0 1 0 0 0 3 0 1 ...
##  $ Parch      : int  0 0 0 0 0 0 0 1 2 0 ...
##  $ Ticket     : chr  "A/5 21171" "PC 17599" "STON/O2. 3101282" "113803" ...
##  $ Fare       : num  7.25 71.28 7.92 53.1 8.05 ...
##  $ Cabin      : chr  "" "C85" "" "C123" ...
##  $ Embarked   : chr  "S" "C" "S" "S" ...
```

As we can see, our response variable (survived) isn't currently a factor. This will cause some issues later down the road, so let's address it now as well as making some other general changes. Of course, I'm changing these just based on 'gut' feeling, so I might have to change them back later on.

```
train$Sex <- as.factor(train$Sex)
train$Survived <- as.factor(train$Survived)
train$Pclass <- as.factor(train$Pclass)
train$Embarked <- as.factor(train$Embarked)
train$SibSp <- as.numeric(train$SibSp)
train$Parch <- as.numeric(train$Parch)


train$Embarked[train$Embarked==""]<-"Any text, NA will be generated"
```

```
## Warning in `[<-.factor`(`*tmp*`, train$Embarked == "", value = structure(c(4L, :
## invalid factor level, NA generated
```

```
train$Cabin[train$Cabin==""]<-NA
```

Let's look to see if there's any missing values.

```
colSums(is.na(train))
```

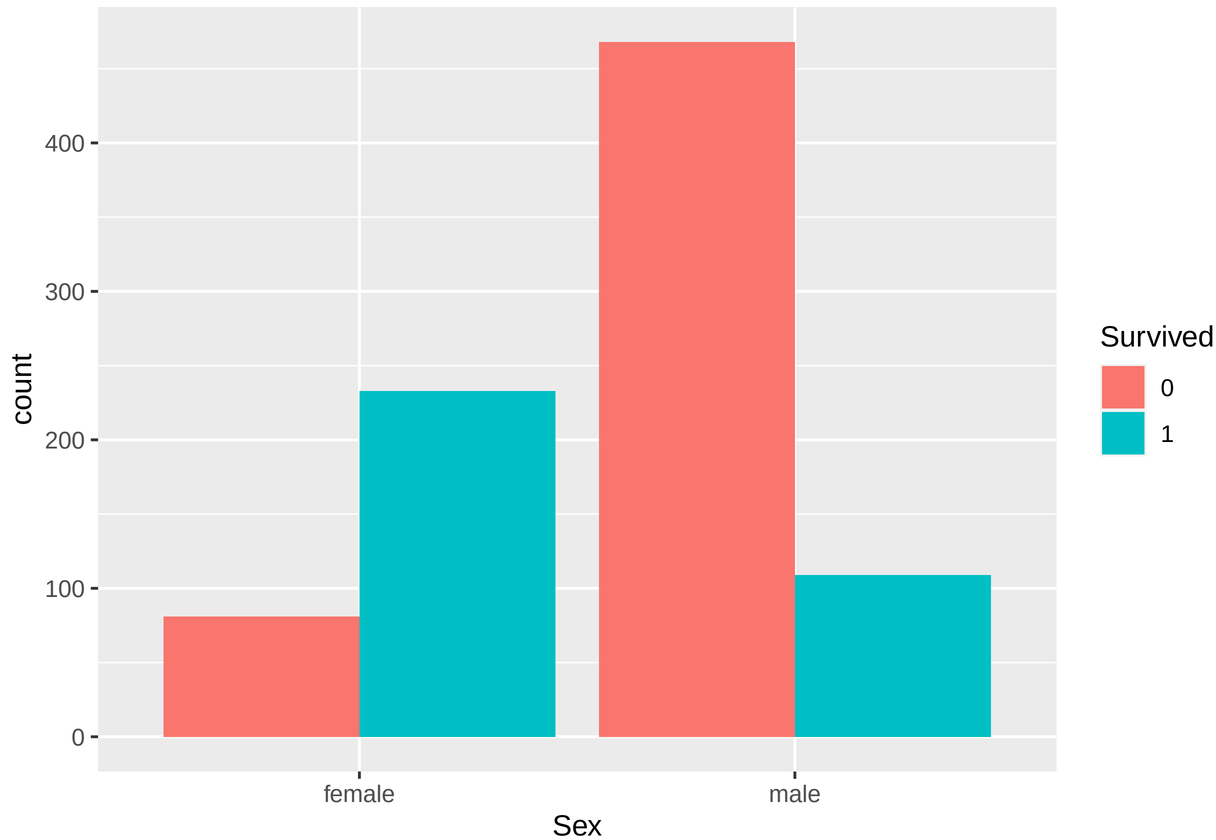
```
## PassengerId    Survived      Pclass        Name         Sex         Age
##           0           0           0           0           0         177
##       SibSp       Parch      Ticket        Fare       Cabin    Embarked
##           0           0           0           0         687           2
```

We've got some to deal with. Let's get into a visual analysis to see if it's worth trying to estimate these missing values.

Visual

To start, let's take a look at age and sex to see how those play into who lives.

```
#Sex
train %>% select(Survived, Sex) %>%
  ggplot(aes(x=Sex, fill = Survived)) + geom_bar(position='dodge')
```



It's clear to see that women survive at higher rates than men do.

```
#Age
train %>% select(Survived, Age) %>%
  ggplot() + geom_histogram(aes(x=Age, fill = Survived))
```

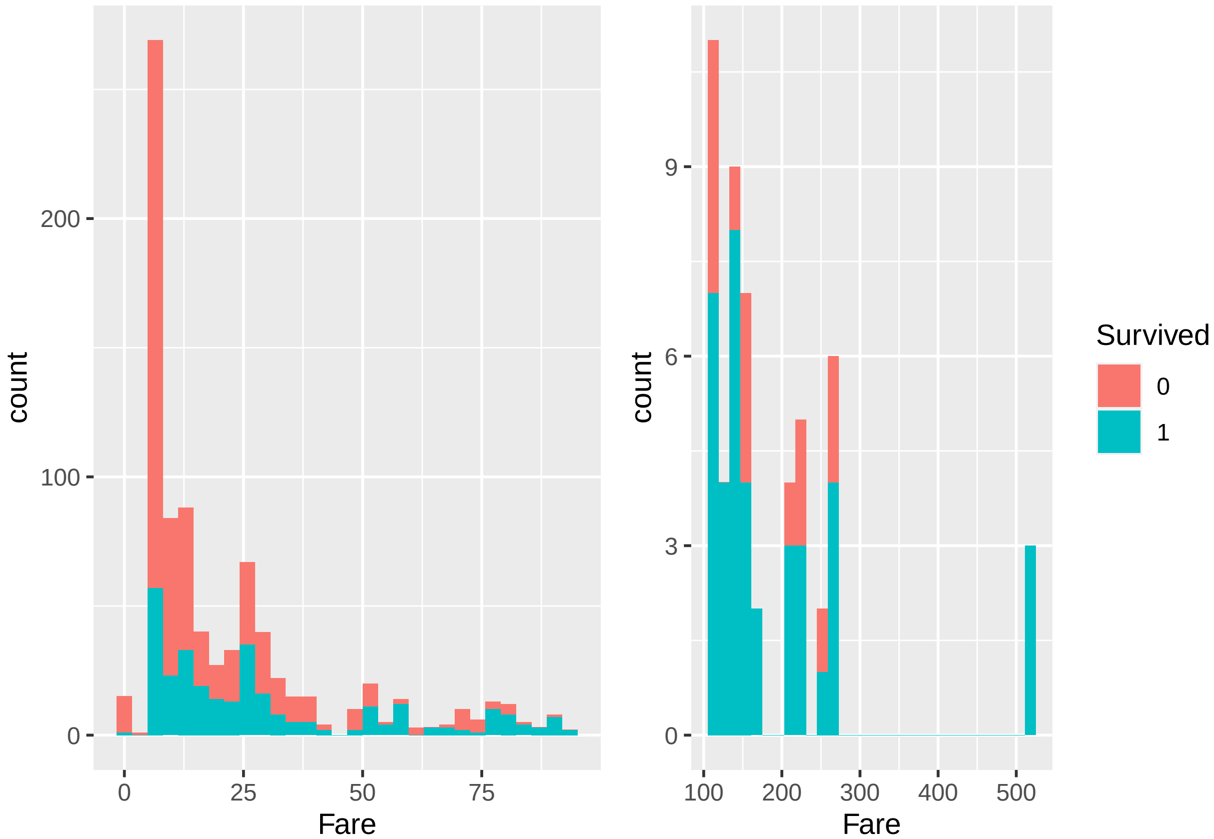
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 177 rows containing non-finite values (stat_bin).
```

Based just on the graphic, it's pretty tough to tell if Age is a predictor. We can try to take a look at economic status based on ticket prices.

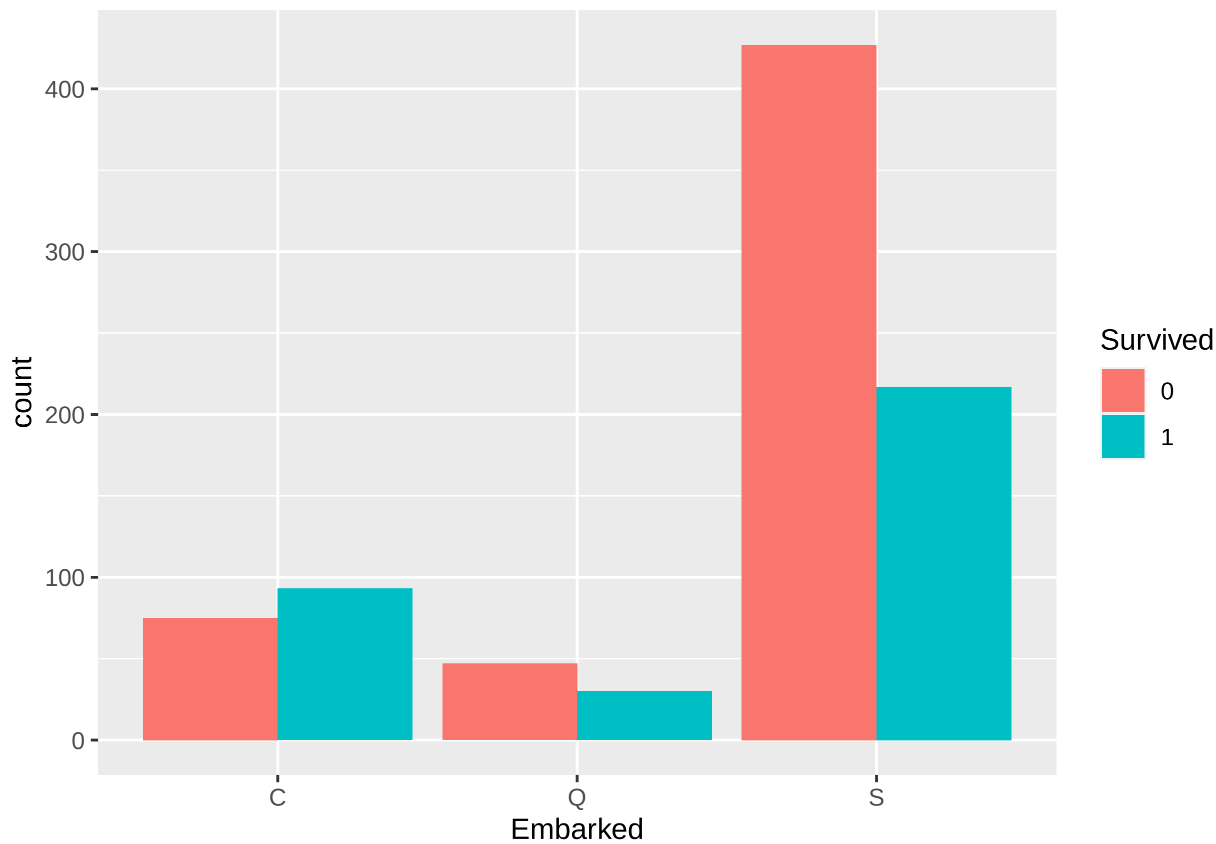
```
#Ticket Price
low.fare <- train %>% select(Survived, Fare) %>% filter(Fare < 100) %>%
  ggplot() + geom_histogram(aes(x=Fare, fill = Survived)) +
  theme(legend.position = 'none')

high.fare <-train %>% select(Survived, Fare) %>% filter(Fare > 100) %>%
  ggplot() + geom_histogram(aes(x=Fare, fill = Survived))

grid.arrange(low.fare, high.fare, ncol=2)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



It's important to note the scales of each graph, but it's easy to see that low priced fares have lower survival rates. With that being said, let's look to see if where people embark from could make a difference.

```
#Embark
train %>% select(Survived, Embarked) %>% drop_na() %>%
  ggplot(aes(x=Embarked, fill = Survived)) + geom_bar(position='dodge')
```



Based on this graphic, it looks like a whole lot more people left from S. Just out of curiosity, let's look into some demographic information. Using both embarked and ticket price may end up overfitting the model, so this step may have some important results.

```
train %>% select(Fare, Embarked)%>%
  group_by(Embarked) %>%
  drop_na() %>% summarise('Median Fare' = median(Fare))
```

```
## # A tibble: 3 x 2
##   Embarked `Median Fare`
##   <fct>            <dbl>
## 1 C                29.7
## 2 Q                 7.75
## 3 S                13
```

This pairs up with the graph above. More people survived than died when embarking from C, and C also has the highest median fare.

```
train %>% select(Sex, Survived, Age) %>% drop_na() %>%
  group_by(Survived, Sex) %>% summarise('Median Age' = median(Age))
```

```
## `summarise()` has grouped output by 'Survived'. You can override using the `.groups` argument.
```

```
## # A tibble: 4 x 3
## # Groups:   Survived [2]
```

```
##   Survived Sex    `Median Age`
##   <fct>    <fct>         <dbl>
## 1 0        female         24.5
## 2 0        male           29
## 3 1        female         28
## 4 1        male           28
```

Here's another interesting conclusion. The median survival age for females is lower compared to those that did not survive. This is the opposite in males. Let's break this down further into the ticket class (pclass).

```
train %>% select(Sex, Survived, Age, Pclass) %>% drop_na() %>%
  group_by(Survived, Pclass) %>% summarise('Median Age' = median(Age))
```

```
## `summarise()` has grouped output by 'Survived'. You can override using the `.groups` argument.
```

```
## # A tibble: 6 x 3
## # Groups:   Survived [2]
##   Survived Pclass `Median Age`
##   <fct>    <fct>         <dbl>
## 1 0        1              45.2
## 2 0        2              30.5
## 3 0        3              25
## 4 1        1              35
## 5 1        2              28
## 6 1        3              22
```

To no surprise, passengers in the highest ticket class had the highest median age. This could lead to the conclusion that age may not be a significant predictor of survived, and ticket class may be more effective. Let's begin building a simple logistic model to start.

Logistic Model

Before we start, we have to take care of some of the missing values. Here's a reminder of what we are missing:

```
colSums(is.na(train))
```

```
## PassengerId    Survived      Pclass        Name         Sex         Age
##           0           0           0           0           0         177
##       SibSp       Parch      Ticket        Fare       Cabin    Embarked
##           0           0           0           0         687           2
```

For now, we'll try out inputting the median age of 28, which is found below.

```
medianAge <- train %>% select(Age) %>% summarise(median=median(Age, na.rm=TRUE))
train$Age[is.na(train$Age)] <- medianAge
```

Now we'll build a couple logistic models.

```r
logistic.train <- train
logistic.train$Age <- as.numeric(logistic.train$Age)

#Changing male to 0 and female to 1
logistic.train <- logistic.train %>%
  mutate(Sex = ifelse(Sex == 'male',0,1))


logistic.train$Sex <- as.factor(logistic.train$Sex)
logistic.train$Parch <- as.numeric(logistic.train$Parch)
logistic.train$SibSp <- as.numeric(logistic.train$SibSp)


str(logistic.train)
```

```
## 'data.frame':    891 obs. of  12 variables:
##  $ PassengerId: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Survived   : Factor w/ 2 levels "0","1": 1 2 2 2 1 1 1 1 2 2 ...
##  $ Pclass     : Factor w/ 3 levels "1","2","3": 3 1 3 1 3 3 1 3 3 2 ...
##  $ Name       : chr  "Braund, Mr. Owen Harris" "Cumings, Mrs. John Bradley (Florence Briggs Thayer)"
##  $ Sex        : Factor w/ 2 levels "0","1": 1 2 2 2 1 1 1 1 2 2 ...
##  $ Age        : num  22 38 26 35 35 28 54 2 27 14 ...
##  $ SibSp      : num  1 1 0 1 0 0 0 3 0 1 ...
##  $ Parch      : num  0 0 0 0 0 0 0 1 2 0 ...
##  $ Ticket     : chr  "A/5 21171" "PC 17599" "STON/O2. 3101282" "113803" ...
##  $ Fare       : num  7.25 71.28 7.92 53.1 8.05 ...
##  $ Cabin      : chr  NA "C85" NA "C123" ...
##  $ Embarked   : Factor w/ 4 levels "","C","Q","S": 4 2 4 4 4 3 4 4 4 2 ...
```

```r
log.mod1 <- glm(Survived ~ Sex + Age + SibSp + Parch + Fare, family="binomial",
                data=logistic.train)
summary(log.mod1)
```

```
##
## Call:
## glm(formula = Survived ~ Sex + Age + SibSp + Parch + Fare, family = "binomial",
##     data = logistic.train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.5065  -0.6526  -0.5426   0.7341   2.3735
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.093214   0.240398  -4.548 5.43e-06 ***
## Sex1         2.623285   0.186341  14.078  < 2e-16 ***
## Age         -0.020583   0.007143  -2.882  0.00396 **
## SibSp       -0.412727   0.102861  -4.012 6.01e-05 ***
## Parch       -0.231918   0.112957  -2.053  0.04006 *
## Fare         0.016396   0.002827   5.800 6.62e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1186.66  on 890  degrees of freedom
## Residual deviance:  850.42  on 885  degrees of freedom
## AIC: 862.42
##
## Number of Fisher Scoring iterations: 5
```

```
vif(log.mod1)
```

```
##      Sex      Age    SibSp    Parch     Fare
## 1.133729 1.107917 1.263037 1.282477 1.170275
```

All in all, the logistic regression model using Sex, Age, SibSp, Parch, and Fare is not too bad. Let's apply these predictions to the data to see how we did.

```
logistic.train1 <- logistic.train %>%
  add_predictions(log.mod1, type = "response") %>%
  mutate(pred = ifelse(pred >= .5, 1,0))
```

Checking confusion matrix and accuracy rate:

```
logistic.train1$pred <- as.factor(logistic.train1$pred)

logistic.train1 %>%
  conf_mat(truth = Survived, estimate = pred)
```

```
##           Truth
## Prediction   0   1
##          0 475 112
##          1  74 230
```

```
## Accuracy rate
logistic.train1 %>%
  metrics(truth = Survived, estimate = pred) %>%
  filter(.metric == "accuracy")
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.791
```

Not the best model at all. Let's explore random forests using the same predictors to see if that will make a difference.

```
train.clean <- train %>% select(-Name, -PassengerId)
train.clean$Age <- as.numeric(train.clean$Age)
RF.1 <- randomForest(Survived ~ Sex + Age + SibSp + Parch + Fare ,
                     data = train.clean, mtry = 2, importance= TRUE)

RF.1 %>% importance
```

```
##              0         1 MeanDecreaseAccuracy MeanDecreaseGini
## Sex   72.51833 98.010211            101.79116        109.22225
## Age   22.95193 24.378700             34.69399         66.28130
## SibSp 30.45931  9.054456             31.39188         24.75961
## Parch 16.65121  8.991918             20.38162         17.10503
## Fare  24.29523 44.400978             47.24720         94.72806
```

Now we'll go head and add these to the data then assess the accuracy.

```
#Adding to data
RF.add1 <- train.clean %>%
  add_predictions(RF.1, type = "response")


RF.add1$Survived <- as.factor(RF.add1$Survived)
RF.add1$pred <- as.factor(RF.add1$pred)


#Assessments
RF.add1 %>%
  conf_mat(truth = Survived, estimate = pred)
```

```
##           Truth
## Prediction   0   1
##          0 526  55
##          1  23 287
```

```
RF.add1 %>%
  metrics(truth = Survived, estimate = pred) %>%
  filter(.metric == "accuracy")
```

```
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.912
```

Just as expected, random forest seems like the model to use. Let's do some more exploration and apply the final model to the testing data and submit.

Let's see which values we need to fill.

```
colSums(is.na(test))
```

```
## PassengerId      Pclass        Name         Sex         Age       SibSp
##           0           0           0           0          86           0
##       Parch      Ticket        Fare       Cabin    Embarked
##           0           0           1           0           0
```

Looks like we'll have to estimate some ages. Based on our EDA from above, we will input the median age depending on where the passenger embarked. This is done to get a slightly better median age which will aid our model more than using the median total age. The same will be done for the missing fare value.

```r
#Age
test.ages <- test %>% select(Embarked, Age) %>%
  group_by(Embarked) %>%
  drop_na() %>%
  summarise("Median Age" = median(Age))

C.age <- test.ages$`Median Age`[1]
Q.age <- test.ages$`Median Age`[2]
S.age <- test.ages$`Median Age`[3]


test$Age <- ifelse(is.na(test$Age) & test$Embarked == 'C', C.age, test$Age)
test$Age <- ifelse(is.na(test$Age) & test$Embarked == 'Q', Q.age, test$Age)
test$Age <- ifelse(is.na(test$Age) & test$Embarked == 'S', S.age, test$Age)

#Fare
which(is.na(test$Fare))
```

```
## [1] 153
```

```r
test[153,]  #Left from S, let's use that median
```

```
##     PassengerId Pclass                Name  Sex  Age SibSp Parch Ticket Fare
## 153        1044      3 Storey, Mr. Thomas male 60.5     0     0   3701   NA
##     Cabin Embarked
## 153             S
```

```r
test.fare <- test %>% filter(Embarked == 'S') %>% drop_na() %>%
  summarise("Median" = median(Fare))
S.fare <- test.fare[1]

test$Fare <- ifelse(is.na(test$Fare) & test$Embarked == 'S', C.age, test$Fare)
```

Boom. Problem solved. Let's apply this to the testing and submit.

```r
test$Sex <- as.factor(test$Sex)
test$Age <- as.numeric(test$Age)
test$SibSp <- as.numeric(test$SibSp)
test$Parch <- as.numeric(test$Parch)
test$Fare <- as.numeric(test$Fare)


RF.test <- test %>%
  add_predictions(RF.1)



#  mutate(prediction = ifelse(pred_pct > .5, 1,0))

final.results <- RF.test %>%
  select(PassengerId, pred) %>%
  rename(Survived = pred)
```

```
write.csv(final.results, 'C:/Users/student/Documents/UVA/Portfolio Projects/generalprojects/Titanic/resu
```